Circuit Drawing Website

FINAL PROJECT REPORT

Team Number: SDMAY19-44

Client: Professor Andrew Bolstad

Adviser: Professor Andrew Bolstad

Team Members/Roles:

Joseph Veal: Back End Code Leader

Alexandra Sutton: Meeting Facilitator, Scribe & Back End Code Designer

Lucas Maring: Report Manager, HTML/CSS Leader

Keegan McCarthy: Team Leader, HTML/CSS Designer

Cassandra Plata: Front End Code Leader

Tyler Schurk: Front End Code Designer

Team Email: sdmay19-44@iastate.edu

Team Website: https://sdmay19-44.sd.ece.iastate.edu

Revised: April 22nd, 2019 / Version: 1.0

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Acknowledgement

We would like to thank our client, Professor Andrew Bolstad, for providing us with this project. We have created a website application that is both an educational tool for students and developed our own programming skills in doing so.

1.2 Problem Statement

Professor Andrew Bolstad teaches Electrical Engineering courses for Mechanical Engineers; however, he is unhappy with his lecture notes concerning circuit diagrams. He is currently using PowerPoint to create circuit diagrams for his lectures, but he found that it is too time-consuming and difficult to create the circuits that he wants. Rather than using pre-made circuit components for his notes, he is using the shapes found in PowerPoint to build his circuits. He wants his circuit components to have thick borders, so they are clearly visible from the grid background, and he wants the wires to stay connected to circuit components when the components are moved around. PowerPoint, however, is unable to function in this way. Furthermore, our client is unable to save the circuits that he creates in order to use the circuit diagram for other class notes or for homework; and so he is continuously building new circuit diagrams. This has been a very time consuming and labor-intensive task for creating his class lectures, so Professor Bolstad is looking for a solution - a website that is intuitive and easy to use to make visually appealing circuit diagrams. He wants a circuit diagram that is neat, easy to read, easy to comprehend, and has labels for each circuit component. In addition, he wants to save these circuit diagrams for later use, efficiently create any circuit he needs in a timely manner, and possibly simulate the circuits for educational practices.

Although there are currently circuit drawing websites available that Professor Bolstad can use, no one website has everything he needs. Therefore, we have created a website application that combines several features of various websites in order to meet every need of Professor Bolstad and his students. We created visually appealing circuit drawings using HTML and JavaScript in order to create a website that can be updated and altered to tailor Professor Bolstad's needs. Our website includes easy-to-use rotate and delete features, it has an "About Components" tab that is an educational tool for the user to learn about various components, it has a "help" tab that explains the program, and the user will have the ability to place circuit components and wiring on a grid.

This website application is not only a tool for students inside Professor Bolstad's class to use, but it is a tool for any students willing to learn about circuit theory. We created both an efficient circuit drawing website and an educational resource that will teach students the fundamentals of circuit theory and application.

2. Requirement Specifications

2.1 Functional Requirements

The functional requirements focus on how the systems should perform.

FR.1: The website allows for the placement of circuit elements, such as wires, voltage sources, resistors, capacitors, and other similar components.

FR.2: Circuit elements should stay connected with a wire if they share the same node and are dragged around the window.

FR.3: Give users the ability to save created circuits for use later, which will require the implementation of user accounts.

FR.4: The schematic G.U.I. should display the basic component information next to its element, i.e. 1kΩ next to a resistor.

FR.5: A user can generate a picture of the created schematic to have it saved and downloadable for use in other documents

2.3 Non-Functional Requirements

The non-functional requirements are primarily focused on the visuals and help portion of the website.

NFR.1: Switchable white/grid background

A button will exist on the website which will allow for a toggleable grid

NFR.2: Help Menu

A separate web page will have information on how to use the website

NFR.3: All Basic Components Available

The application will feature common electrical components that a user will see in a general electrical engineering course

NFR.4: Circuits need to be easily legible

Circuits need to be legible within presentations and reports

NFR.5: Educational tab with example circuits

An educational tab will feature equations and behavior of the electrical components that can be seen in the drawing tool.

2.3 Use-Cases

This website will primarily be used by our client, Professor Bolstad, but it will also have the potential to be used by any students at Iowa State who are interested in learning more about circuit theory. Users will be able to create and download circuits for lecture notes, homework, or labs in order to incorporate visually appealing circuit drawings anywhere. The website also has an educational tab that teaches users about various circuit components that are available to use in the website. Furthermore, any outside user will not be able to alter any of the code associated with the project, so the user will not be able to harm the website in any way. All user accounts will be securely stored within the database on the server that the website is used on.

3. System Design and Development

3.1 Design Plan

Due to our limited experience with the creation of web applications, our initial work had consisted of research and brainstorming. One of our first tasks was to take a look at competitors, both paid and free, to see what the positive and negative features were of each. We analyzed the applications that we have had to use inside and outside of our classes such as PSPICE, Multisim, and Falstad. PSpice is a tool used for the simulation of circuits while Multisim functions as both a drawing and simulation tool. These are heavy, expensive tools that are quite a bit beyond our end-goal for the drawing tool but they offered insight into how accessible we should make our website. Falstad is a free web application that some of our teammates have used in the past that functions similarly to Multisim. Its ease of use was an important takeaway; however, the circuit drawings are not visually appealing to place into documents.

Moving forward with our project required us to decide what programming language we wanted to use. Python and JavaScript were the two primary choices due to prior experience. They both feature extensive packages that would be beneficial for use on our website. With our research, JavaScript seemed appropriate because of the drawing packages available. However, Python is easier to learn because of prior experience with C coding, while only one member is familiar with Java. Our group saw the advantages with both so we decided to meet in the middle and use JavaScript for the front end functions and Python for the back end functions.

The creation of the website was split up into three categories: HTML/CSS, JavaScript front end functions, and Python/Django back end functions.

The HTML/CSS code provides the basic visuals for the site. This includes all the area surrounding the main canvas element in the center of the web page.

JavaScript is used to create the functions for the interactive portion of the website. These functions allow for components to be placed, dragged, and snapped within the drawing space. We needed to find a drawing library that could accomplish the listed features and functional requirements. There were a couple of open source drawing libraries we tested that allowed for this type of functionality, but eventually settled on using the Fabric.js library after testing.

Our website made use of the Django web framework to manage all back end functions. Anytime a request is made to the server, Django matches the URL of the request to a view. A view determines

the type of request and generates a response. The response contains any necessary database information and an HTML template. The view also handles back end processing. The view is where we chose to create our PNG files from the requested data. Django handles all routing, database interaction, and all rendering. Django has a large number of modules available to easily implement difficult web development features. Our site makes use of Django Modules for user authentication and user session management. The features of Django allowed us to create advanced website features in a secure and managed environment. Django evaluates all content and requests the server receives to prevent malicious content from being uploaded. This proved to be a major developmental advantage for our group.

3.2 Design Objectives

The main design objectives that lead to our project being considered successful are the following:

- Toggleable grid
- A wide variety of components snapping to the grid
- Component Labeling
- Wiring tool
- User Login and Logout
- User Download and Save
- Saved Designs Tab

With one of these design objectives not functioning, it would lead to the whole project being considered unsuccessful, however, all of these design objectives were achieved and are working as intended.

3.3 System Constraints

We did not have many system constraints during our project due to our project being web-based, therefore, the website does not require anything that would require a constraint.

Our client has talked about having a future senior design team continue the work on our project, so our code needs to be well commented and accessible to our client. We used GitLab for the integration of our code and the version management of our code.

3.4 Design Trade-Offs

Every component image was created to function on a grid where the origin is at a specific point, such that the leads of the components are in line with where the wires can be placed. One feature of our drawing tool was the labeling of components as they serve as an identifier and value for the component, i.e. R1 = 100 Ω. Our first designs had the label added and grouped together with the addition of the component. This caused the object's origin to move position as what was originally two images, the component, and label, are now considered one image. Whenever the component/label combination was moved around the space, the leads no longer lined up with the grid. To fix this, labels are no longer added at the same time as the components. Rather, a separate

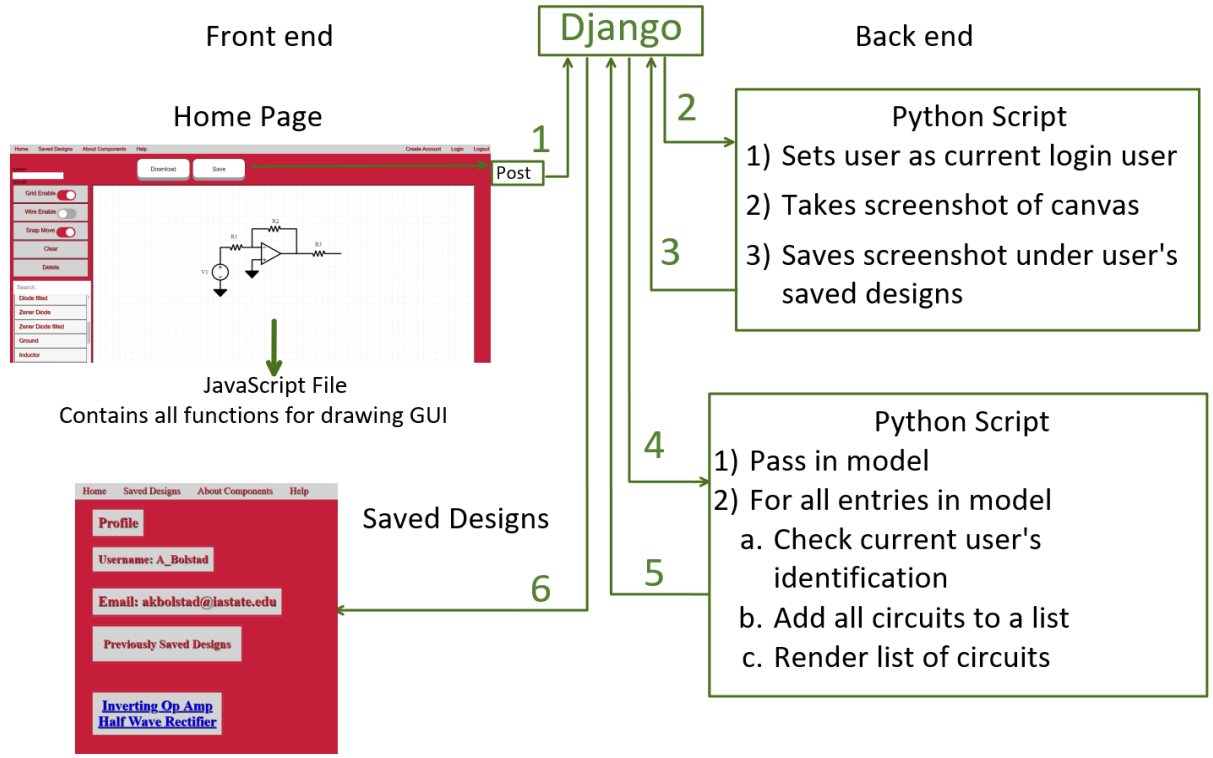
button is used to add a label. This adds an extra step in the circuit creating process but makes dragging components around the drawspace much easier.

The website is intended for ISU servers, as opposed to an Amazon or Microsoft server. It requires a dedicated virtual machine with web access. ISU virtual machines have specific amounts of memory, but we are allowed to access them for free. The memory restriction is important because users of our website save images to our machine. Django's file field for a SQL database is just a reference to a local file, so all images must be saved to the virtual machine. There would be a cost associated with a server on Amazon or Microsoft services, but we would be able to expand the memory as needed. Another tradeoff with an ISU server is the amount of RAM that our website will have. We keep track of user sessions using cookies. These cookies are stored in RAM. If the website were to gain popularity, an ISU virtual machine may not be able to support the website.

3.5 Architecture Diagram and Design Block Diagram

The diagram below generally outlines the structure in which our website interacts between the front end and back end. As a user interacts with any kind of circuit drawing, it is handled purely by the front end with the use of JavaScript functions. As the user attempts to save and/or download their finished image, the figure below outlines how the Django framework handles the user authentication and passing of information.

Figure 1: Website Architecture and Django



4. Implementation

4.1 Implementation Diagram and Technologies Used

To manage our code, our team used Git. Git allowed us to easily control versions of our project and a code manager had to approve changes before they were made to our main repository. Git is a commonly used and free tool that can be downloaded on any operating system. Each team member has a Git repository on their computer that interacted with an online repository on GitLab.

Our team also made use of Django as previously described. Django completely managed our website back end and helped us accomplish the following:

- Routing
- HTML Rendering
- Database Interactions
- User Authentication and sessions
- Data processing

4.2 Rationale for Technology and Software Choices

HTML is an important tool in any web development project. HTML allows a developer to design the basic layout of a website. It has specific tags that are used to determine the section of a website for a piece of text. It is also used to add links and many other basic functions of a website. In our case, HTML exists inside of a Django file known as a template. The file is then rendered in the browser of the user once the Django view has finished all back end processing.

Our team felt the best option to manage our back end was Python because Python is an easy to learn language. Python is an interpreted language which makes it difficult to debug at times; however, it does not require a compiler which was advantageous for our group. This is an advantage to us because it allows Python to function as a scripting language. Python also has a wide range of modules that our team used to efficiently run backend functionality.

The use of the Fabric.js drawing library was an easy choice after reading its documentation and testing the pre-built demos. A standard canvas will allow the placement of images but has limited interaction. Fabric heavily expands the capabilities of the HTML Canvas and drastically simplifies the code. Your average electrical / software engineer should be able to look at the code and more easily understand what a function is accomplishing. This makes the creation and modification of the drawing systems a much simpler process compared to doing so without the library.

4.3 Applicable Standards and Best Practices

A project of this caliber most often includes functional requirements and nonfunctional requirements. In many cases, these requirements are given by the user. Some examples of these types of requirements are protocols and security standards. This project has given our team lots of freedom to choose protocols and procedures that we must adhere to. On the other hand, since we

are doing a web development project, there are certain protocols that we must follow in order to have our application operational.

One of the protocols that are required for our project is the HTTP protocol. This is a standard web protocol that allows for web development languages to be sent as data over the internet. The HTTP protocol helps us with verification. HTTP has a variety of status codes that allowed us to debug issues by reading the server log. Another protocol our team needs to keep in mind is the IP protocol. This protocol will be important when we are testing and setting up the server. The IP protocol assigns our server an address and allows other machines to make HTTP requests to it. Once we migrate our server we will have a permanent IP address, but while we are in testing we will take advantage of one of the localhost ports for our HTTP requests. These protocols are more important for networking than for web development; however, they must be followed if our project is to work correctly. To automatically handle these protocols, we have investigated web development techniques that will help us meet our nonfunctional requirements.

Our group did extensive research to determine which coding languages and frameworks we should utilize to develop the best version of our project. Our conclusion led us to use Python and the Django framework to manage the back end, and JavaScript to create a GUI for our front end. Using a framework allowed us to control the database and any routing that the web server we will be using. The management of routing allows us to adhere to the standards of the IP protocol and helps us achieve our nonfunctional and functional requirements. Simply choosing standard web development languages will allow us to use the HTTP protocol correctly.

Our team was required to follow security protocols. A simple and very effective protocol that will allow us to protect our website is the OAuth protocol. Once our group determined that this protocol was needed, signing into the website is required to access the content. In order to sign into the website, the user needs to present a valid username and password. The Django framework also allows us to use modules to perform security functionality and the framework does have an OAuth module available

When it came to determining the best way to layout all of the various requirements needed for the website we used the following standard, “IEEE 830-1998 - IEEE Recommended Practice for Software Requirements Specifications.” This standard lays out different ways to gather the software requirements and make plans on meeting these requirements. This was vital for the beginning stages of the project as it allowed the group to communicate with Professor Bolstad more effectively.

5. Testing, Validation, and Evaluation

5.1 Test Plan

Functional Testing

The following tests have been separated into three categories. Those categories are unit tests, box tests, and system tests. Each test case includes a procedure and status. These cases are designed for basic layout and design.

5.2 Unit Testing

Unit Tests *(Table 1)*

Number	Test	Desired Outcome	Status
1	Page Links	Clicking links yield HTTP 200 Status Code	Success
2	GUI Drag and Drop	Dragging a component places it on GUI	Success
3	GUI Wires	Wires must move with the object when the object is dragged	Failed
4	GUI Background	Background must be clients desired color	Success
5	GUI Component Menu	GUI component menu can place parts into GUI and all parts are easily visible	Success
6	GUI Hotkey Placement	Pressing one of the Hotkeys displays a component on the GUI	Not Implemented
7	Download Button	Download button results in a file that matches the user's design	Success

8	Save Button	A record of the current circuit can be found in the database	Success
9	Help Menu	Help menu appears when clicked, HTTP 200 Status code returned	Success
10	Sign in	Sign in allows for authentication, rejection, and user creation	Success
11	New Page Rendering	Django server displays the result of last HTTP Request	Success
12	Database	Database features can be saved and accessed in HTML templates	Success

5.3 Interface Testing

Box Tests *(Table 2)*

Number	**Test**	**Desired Outcome**	**Status**
1	GUI	Schematics can be created with all parts labeled and mobile	Success
2	Authentication	Correct Username and Password allow for the viewing of old schematic	Success
3	Homepage	The homepage allows for GUI creation, site navigation, and download capabilities	Success

4	Backend	The back end is able to process any data necessary and control website views	Success

5.4 System Integration Testing

The system integration testing for this project was minimal seeing as though it was all web development. Our main concerns with this project involved integrating the JavaScript front end functionality with the Django server, and the communication with the back end. Throughout the development of the features of this website, we used Google Chrome but also ensured that the libraries, layout, and functionality would also be working with any web browser that a user may encounter.

5.5 User-Level Testing

We believe that the most effective test results are in the form of user feedback. Our team obtained feedback from several sources and the following paragraphs will describe our process for collecting user feedback, the positive user feedback and the negative user feedback.

To collect feedback from users, we asked our users to attempt to make a circuit with no knowledge of the website. We asked them to make accounts and save a circuit to their account as well. We started by giving very little help with creating a circuit to see where they encountered difficulties and confusion. We then provided them feedback when they became stuck, so we could see how other they liked other parts of the website as well. This process helped us develop our “help” tab because this tab gave a general outline on the best way to create a circuit. We tested several students in the TLA and our client. Their feedback will be described in the following paragraphs.

Some of the positive feedback we received was that our drawspace was easy to see and our components were easily visible. Users said that our site was easy to navigate and contained all the necessary components. They thought that our implementation of sessions made it easy for them to access older circuits. Users enjoyed our pleasing design and our component placement features as well. Our site was kept organized by having only essential features, which reduced user confusion.

Moreover, we received some constructive feedback from our users on how the website could be improved in the future. The users struggled to grasp how to use some of our features without an explanation. They thought an instruction help page was needed to get started. We also noticed that they had a hard time understanding how to use our snap feature. To combat this issue, we created a process that will produce high quality, presentable circuits. Some other user’s feedback involved the background color of the images being saved. When previewing a circuit in chrome, there is no white background and that makes the circuit very difficult to see. Users would have also liked to be able to edit circuits at a later time. We currently have accessible PNG files of their saved schematics. These changes will help us improve the project in the future.

5.6 Validation and Verification

Numerous team members, as well as our advisor/client, Professor Andrew Bolstad had agreed that our unit tests had passed. We noted that the next part of our validation and verification needed the approval of other users.

We believe, as mentioned above, that the user's validation in the effectiveness of our final product is pivotal, and therefore we worked to try to implement what both we and our peers and users would want in a circuit drawing website. We had implemented as many of the features that we could and passed our unit testing criteria to both our team's, our client's, and our peer's satisfaction.

5.7 Evaluation

The final product that we have ended up with allows us to believe that we have a functional final product, but there can be improvements made for an easier user experience. After initial testing, and realizing that the best way to create the circuit was through a specific process, we had worked to alleviate the choppy experience with some fixes with the drawing GUI. After fixing prominent issues that we have seen, we recognize that the user experience can be made easier with less mouse moving if we had created hotkeys for the user to generate wires, and delete components.

We also recognized that we did not meet all of our stretch goals of simulating, auto wire remapping, and editable saved designs.

6. Project and Risk Management

6.1 Task Decomposition and Roles and Responsibilities

In the early planning stages for the project, we split up into three groups of partnerships to work on different parts of the project. Tyler and Cassie were tasked with working on the front end and drawspace design. Alex and Joe worked on the back end and Django server work. Keegan and Luke worked on the HTML/CSS part of the project.

As time went on throughout the project life, there were different variations of the groups and some team members spent time working with other groups. This allowed for the needed flexibility to put the necessary work hours towards different parts of the project that needed it.

Furthermore, below are the individual responsibilities for each member of the project:

Keegan: Team Leader, HTML/CSS Designer

- Creating and proposing various different schedules
- Working on HTML/CSS
- Creating all on the vector copies for the components used
- Developing JavaScript functions for each component to lock on to the grid properly

Alex: Meeting Facilitator, Scribe, Back End Code Designer

- Communicating with Professor Bolstad
- Setting up meetings each week and reserving rooms
- Developing and designing back end user authentication

Luke: Report Manager, HTML/CSS Leader

- Creating weekly status reports for the group
- Uploading important files to the project website
- Designing HTML/CSS part of the website

Cassie: Front End Code Leader

- Designing front end functionality
    - Wiring tool
    - Snappable objects
    - Drag and drop
    - Component Labeling
- Migrating all files to the master branch

Tyler: Front End Code Designer

- Designing front end functionality
    - Wiring tool
    - Snappable objects
    - Toggleable grid
    - Rotation lock
    - Drag and drop
    - Component Labeling

Joe: Back end Django Code Leader

- Developing back end Django server setup
- Creating Django views to control file service and back end processing
- Working on image saving feature
- Creating tutorials to help each group member run the server locally

6.2 Project Schedule

Table 4: 1st Semester Schedule

1st Semester			DURATION (days)
START DATE	END DATE	DESCRIPTION TITLE	
9/23/18	9/29/18	Complete Project Plan version #1; Determine Coding Languages to be used	6
9/30/18	10/6/18	Meet with Andrew Bolstad; Learn the basics of Python, JavaScript, HTML, and CSS	6
10/7/18	10/27/18	Learn Python, JavaScript, HTML, and CSS	20
10/28/18	11/10/18	Prototype of the website, using HTML and CSS; Back end development	13
10/28/18	11/10/18	Back end development Django	13
11/11/18	12/1/18	Front end GUI development. Circuit component drawing, drag, and drop	20
12/2/18	12/15/18	Complete prototype I, Testing with Professor Bolstad and random users; Gain feedback and make improvements to Prototype I	13

Figure 6: Semester 1 Gantt Chart

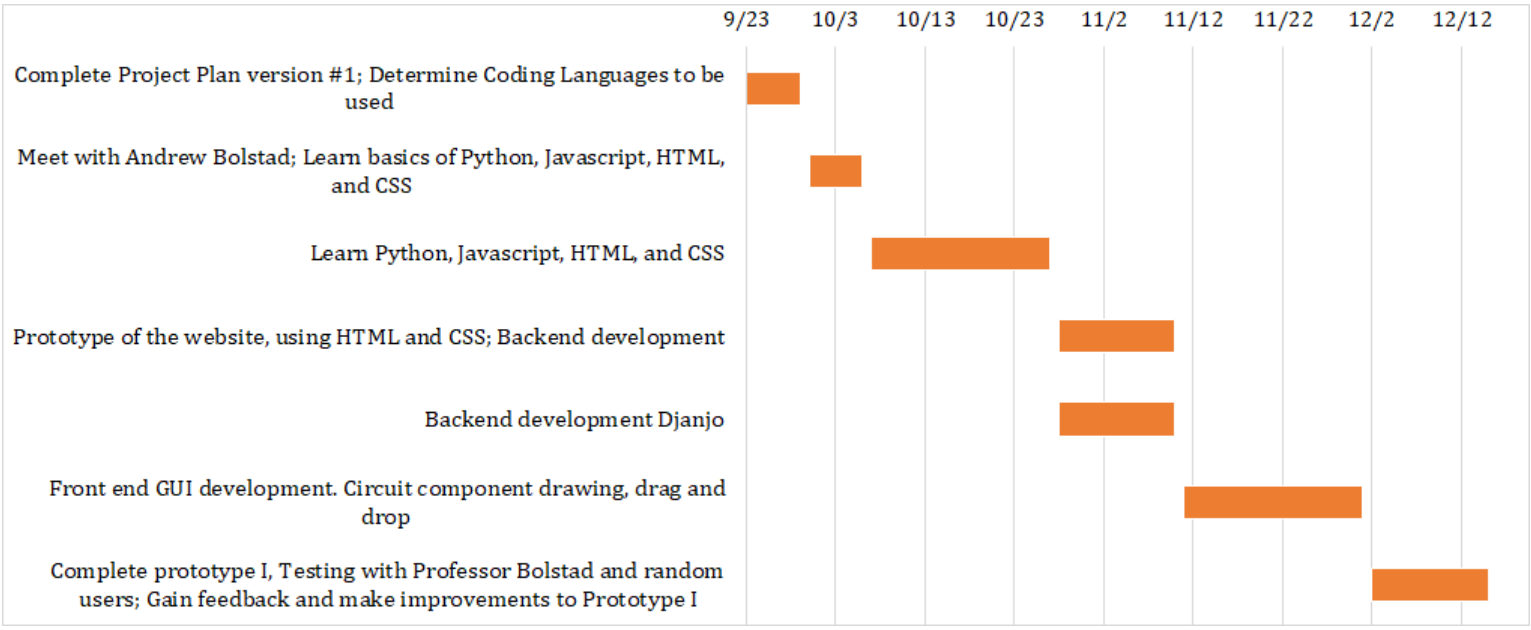


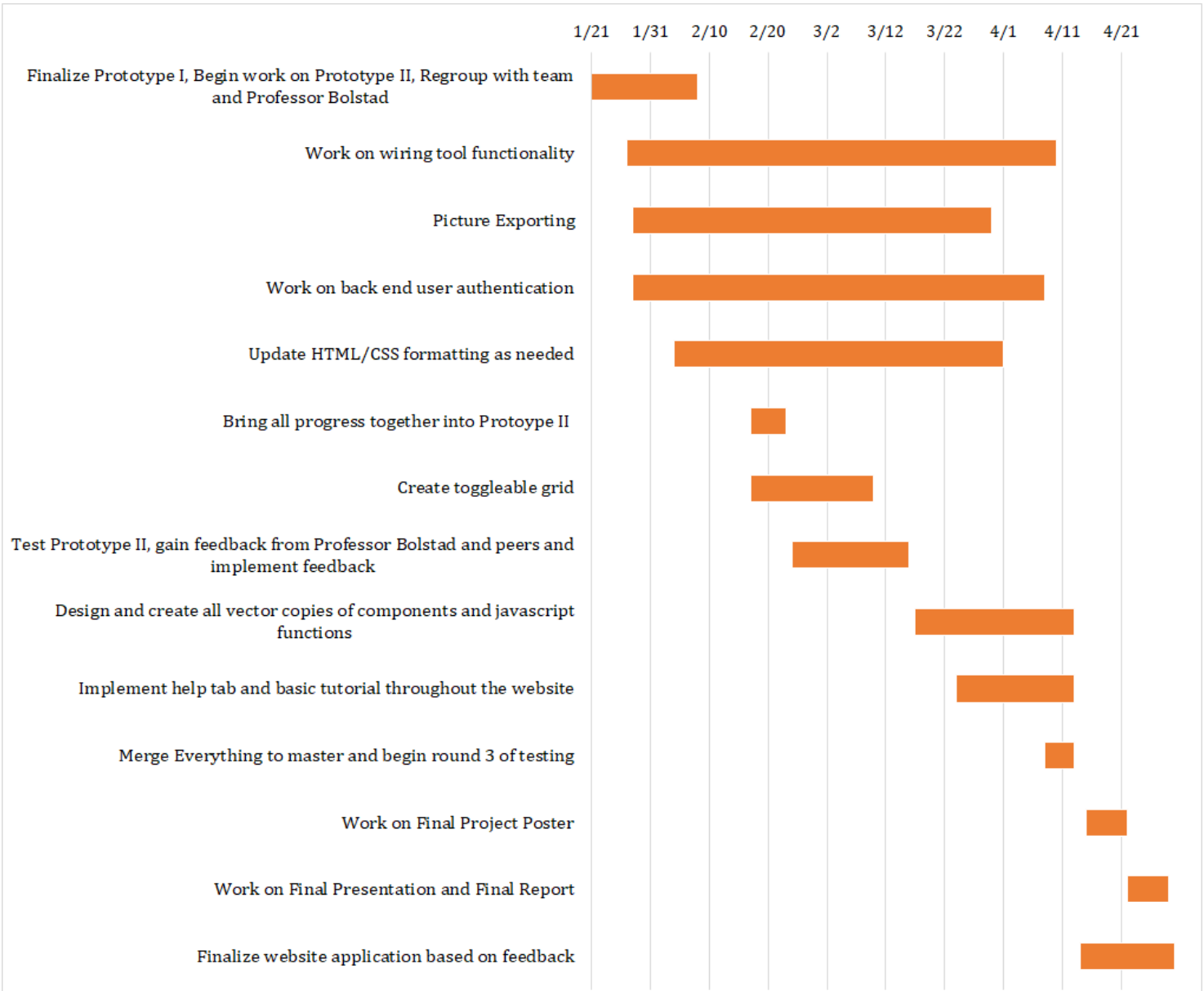
Table 5: 2nd Semester Schedule

2nd Semester			DURATION (days)
START DATE	END DATE	DESCRIPTION TITLE	
1/21/19	2/8/19	Finalize Prototype I, Begin work on Prototype II, Regroup with team and Professor Bolstad	18
1/27/19	4/10/19	Work on wiring tool functionality	73
1/28/19	3/30/19	Picture Exporting	61

1/28/19	4/8/19	Work on back end user authentication	70
2/4/19	4/1/19	Update HTML/CSS formatting as needed	56
2/17/19	2/23/19	Bring all progress together into Prototype II	6
2/17/19	3/10/19	Create a toggleable grid	21
2/24/19	3/16/19	Test Prototype II, gain feedback from Professor Bolstad and peers and implement feedback	20
3/17/19	4/13/19	Design and create all vector copies of components and JavaScript functions	27
3/24/19	4/13/19	Implement the help tab and basic tutorial throughout the website	20
4/8/19	4/13/19	Merge Everything to master and begin round 3 of testing	5
4/15/19	4/22/19	Work on Final Project Poster	7

4/22/19	4/29/19	Work on Final Presentation and Final Report	7
4/14/19	4/30/19	Finalize website application based on feedback	16

Figure 7: Semester 2 Gantt Chart



6.3 Risks and Mitigation

A big concern that hindered our plan is the fact that we are creating a web application with coding languages Python, JavaScript, and HTML. As a team consisting of all Electrical Engineering majors, we do not have a lot of experience using these languages. As a result, our project team needed to put in the additional time learning these languages on our own time. While we were confident that we can design the circuit drawing web application using these languages, we needed to plan for the time needed to both learn the languages and complete the project.

Furthermore, the long term feasibility of achieving a fully functional circuit simulator was certainly a stretch goal from the beginning. Proper circuit simulation will be a lengthy process and will be dependent on the progress of the circuit drawing functionality. The circuit drawing functionality is the most important part of the project and will be completed in its entirety before working on simulation. It is concerning that the website application will be incomplete if incapable of simulation; however, this might be something another senior design team can complete in the future.

Our last main risk is the lack of security implementation. By the end of the project, we implemented basic security into the website, which lead to Iowa State raising concern to make the website public. Further security measures would prevent malware from harming Iowa State's servers. This is something that our group was not able to accomplish and that a future senior design team could implement.

6.4 Lessons Learned

After working on this project for the past two semesters we learned various lessons. The first lesson, being that we needed to stick to our proposed schedule more effectively. There were times when things fell behind due to workload in other classes that we are taking. As a result of not sticking to the schedule on time, certain functionality is not perfected to the extent that we may have liked.

The second lesson learned would be having a group of all electrical engineers working on a software engineering project. This made the whole project a lot more difficult due to each group member having to self-teach themselves different coding languages to get the project done. Looking back on it now, we realize this was not the best decision to pick this project, however, we are still proud of what we accomplished while self-teaching ourselves the whole time.

7. Closure Material

7.1 Conclusion

There exist many different circuit applications, but they are often weighed down with heavy processor requirements and expensive licensing costs. In addition, some of these applications are not free and intuitive to use. Therefore, we created an educational, easy to use, and free web-based application that will eliminate these issues. Our website features an appealing circuit drawing tool

that can be used to teach students who are interested in circuit design and theory. We also added educational features that allow students to better understand the circuits they created using our website.

We divided the work accordingly so each member learned how to utilize both Python and JavaScript. This gave each member thorough experience with software design, and it allowed everyone to follow the progress of the project. Furthermore, we had weekly meetings in order to discuss the team's accomplishments, schedule, and issues concerning the various components of the project. This constant communication ensured an efficient design process and it allowed us to stay on top of any issues in order to resolve them as soon as possible. Although we are all electrical engineering students, our planning and organization skills are exactly what this project needed to solve this software design problem.

We believe that we accomplished a lot as a group these past two semesters, completing many of the deliverables set at the beginning of the project. As all projects go, there was more to be accomplished than we actually completed, which is talked about in the coming sections.

7.2 Closing Remarks

Despite being a group of six electrical engineers, each member was able to learn how to reach the goals that we have set for the project. We learned various new skills as well expanded on previous skills acquired throughout our college career in order to create this website from scratch. As a team, we have completed the primary requirements set by Professor Bolstad and hope that this final project is exactly what he needed. Even though improvements and additions can be added to the final project, the circuit drawing website is a very functional tool that any user can experiment with to learn more about circuit theory.

7.3 Future Work

In the above section in which we discussed the Evaluation of our Testing portion, the last note that we recognized was that there were stretch goals that we could not meet in time. The future work for this project may include the ability to simulate finished circuits, the ability to edit old saved designs, and integration with the React front end framework for a cleaner and more visually appealing and dynamic website aesthetic, as well as wire auto-routing. These tasks can be carried on to future senior design teams moving forward.

Below is a demo link for our latest working version of our project:

https://youtu.be/Fp9jIvC19Cg

8. References

"Django." *Documentation for Django.contrib.auth.forms*, docs.djangoproject.com/en/1.8/_modules/django/contrib/auth/forms/.

"Django." *Documentation for Django.contrib.auth.models*, docs.djangoproject.com/en/1.8/_modules/django/contrib/auth/models/.

Smith, Caleb. "Learning Django." *Lynda.com - from LinkedIn*, Lynda.com, 22 Jan. 2018, www.lynda.com/Django-tutorials/Learning-Django/656811-2.html.

IEEE 830-1998 - IEEE Recommended Practice for Software Requirements Specifications

9. Team Information

Luke Maring - HTML/CSS

Email: Luke.maring43@gmail.com

Keegan McCarthy - HTML/CSS

Email: keeganmccarthy27@yahoo.com

Cassie Plata - Front End

Email: cassieplata@gmail.com

Tyler Schurk - Front End

Email: tylerschurk@gmail.com

Alex Sutton - Back End

Email: asutton512@gmail.com

Joe Veal - Back End

Email: joeveal33@gmail.com